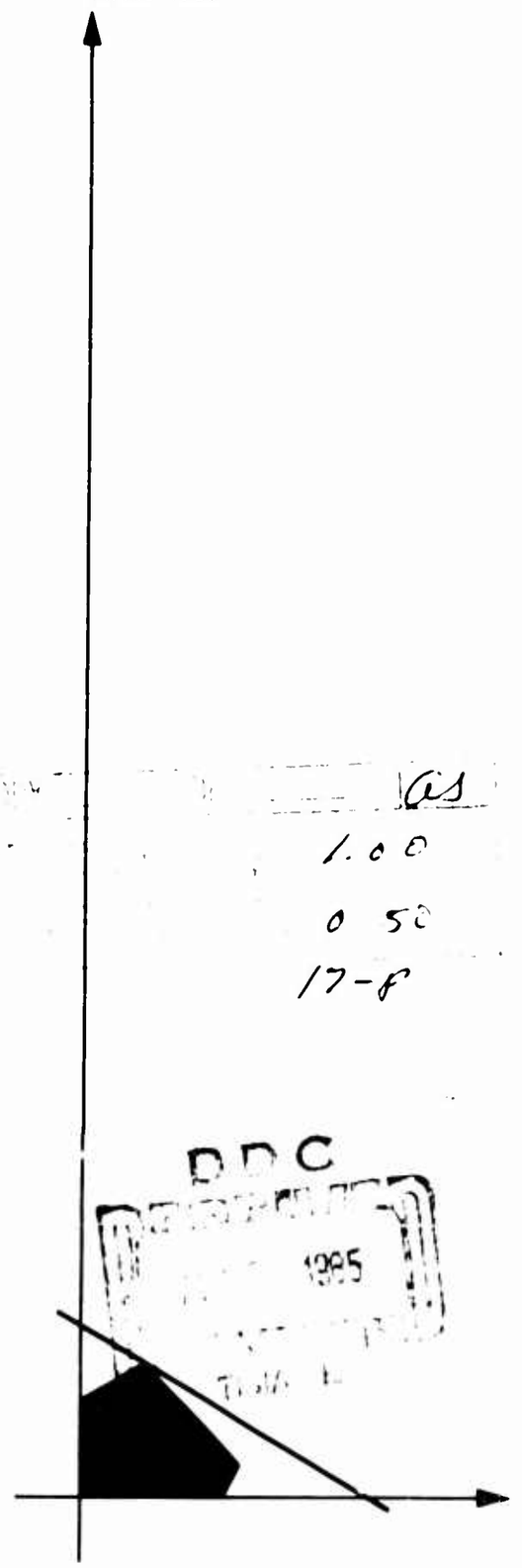
ORC 65-11
APRIL 1965

MULTI-TERMINAL SHORTEST PATHS

by

T. C. Hu



OPERATIONS RESEARCH CENTER

COLLEGE OF ENGINEERING

UNIVERSITY OF CALIFORNIA - BERKELEY

MULTI-TERMINAL SHORTEST PATHS

by

T. C. Hu
Operations Research Center
University of California, Berkeley

April 1965 ORC 65-11

This research has been partially supported by the Office of Naval Research under Contract Nonr-222(83) and the National Science Foundation under Grant GP2643 with the University of California.

ABSTRACT

The present paper gives an algorithm that finds simultaneously the shortest paths between many pairs of nodes in a given network. In the book by Berge, the values of shortest paths between many pairs of nodes are found. Here, we use a special matrix multiplication technique to find the actual arcs that are used to form the shortest paths. In a network with n nodes, $\log_2[n-1]$ special matrix multiplications are needed to find all the shortest paths.

The present paper also gives an algorithm for constructing a network with prescribed shortest distances and with the total distances associated with arcs a minimum.

The present paper gives an algorithm that finds simultaneously the shortest paths between many pairs of nodes in a given network, an algorithm for constructing a network with prescribed shortest path distances, and with the total distances associated with arcs a minimum.

Many papers have been written about finding one shortest path from a given node to another node in a given network (see for example, [1], [3], [4], [8], or the review [9]). If we are interested in shortest paths between many pairs of nodes, then this problem can certainly be solved by using any of the existing algorithms for finding a shortest path and doing the algorithm over and over for every pair of nodes. Because in finding a shortest path many informations are obtained during the process, we would expect that some saving would be acheived if we want to find many shortest paths. Also, for finding maximum flow values between many pairs of nodes, it is known that a great amount of computation can be saved (see [5]), and the known dual relationship between maximum flow and shortest path in an a-b planar network would also lead one to expect that many shortest paths can be found simultaneously. In the book by Berger [2], the values of shortest paths between many pairs of noeds are found by a special matrix multiplication technique; in Gomory and Hu [6], a minimal cost network is found which can satisfy given flow requirements between many pairs of nodes simultaneously.

We are now interested in finding the actual arcs that are used to form the shortest paths. Although the paper is self-contained, we shall use many of the notions of [2] and [6].

We consider a network consisting of nodes N_i and arcs A_{ij} leading

from nodes N_i to N_j. Associated with each arc, there is a distance function d_{ij} defined on it. We shall not assume either $d_{ij} = d_{ji}$ or that $d_{ij} + d_{jk} \geq d_{ik}$, but we shall assume throughout the paper that distance functions d_{ij} are all non-negative.

We shall follow many authors (see for example, [2]) in defining two binary operations

$$\lambda_1 \textcircled{+} \lambda_2 = \min(\lambda_1, \lambda_2) \tag{1}$$

$$\lambda_1 \textcircled{\times} \lambda_2 = \lambda_1 + \lambda_2 \tag{2}$$

Using this special addition and multiplication instead of usual addition and multiplication, we have a special matrix product of the two $n \times n$ matrices $D_1 = [d_{ij}]$ and $D_2 [d'_{ij}]$ the matrix $D_3 = [d''_{ij}]$ with

$$d''_{ij} = \min[d_{is} + d'_{sj}] \quad s = 1,\ldots,n . \tag{3}$$

If we square $D = [d_{ij}]^2$, the distance matrix in the above sense, the resulting matrix $D^2 = [d''_{ij}]$ would give the values of shortest paths using two or less arcs between any two nodes. Similarly, D^{n-1} would give the shortest path values between any two nodes using $n-1$ or less arcs. Since, for a network of n nodes, shortest paths are always of $n-1$ arcs or less, after L times operations like (3), D^{2^L} is obtained with $2^L \geq n-1$ and $L \leq [\log_2(n-1)]$ where $[\log_2(n-1)]$ indicates the least integer greater than or equal to $\log_2(n-1)$.

In the paper by Gomory and Hu [6], the problem is to find a minimum cost network that will satisfy all given flow requirements simultaneously. Assume the cost of building an arc of unit capacity between nodes N_i and N_j is c_{ij}; we want to find y_{ij} such that

$$(4)\quad \sum_i x_{ij}^{pq} - \sum_k x_{jk}^{pq} = 0 \qquad j \neq p,q$$

$$(5)\quad R^{pq} = \sum_i x_{ij}^{pq} \qquad j = p$$

$$(6)\quad \sum_{p,q} x_{ij}^{pq} \leq y_{ij} \qquad \text{for all } i \text{ and } j$$

and

(7) min $z = c_{ij}y_{ij}$ where R_{pq} are given flow requirements between nodes N_p and N_q. This problem can certainly be solved by using c_{ij} as distance and then find the shortest paths between each pair of nodes N_p to N_q and give every arc in the shortest path the amount of $y_{ij}^{pq} = R^{pq}$. The final y_{ij} is obtained by addition $\sum_{p,q} y_{ij}^{pq} = y_{ij}$. Instead of finding shortest paths one by one, we can find the cheapest network as follows: (see [6]). We first form the $n \times n$ matrix $D = [d_{ij}]$ where d_{ij} is the distance from nodes N_i to N_j. Then we form the powers of D by squaring in the sense of (3) until D^{2^k} is obtained where $2^k \geq \log_2(n-1)$.

In getting D^2, D^4, D^8, etc., we need only keep the largest power of D and discard the previous powers of D. At the same time, we form matrices $B_1, B_2, \ldots, B_k$ where $B_u = [b_{ij}^u]$ and b_{ij}^u is the index s in (3) for which the minimum was obtained.

After $\log_2(n-1)$ operations or less, we have D^{2^k} and B_1, B_2, ..., B_k.

Now we define another set of matrix $\bar{B}$ from B_k. We start with $\bar{B}_{L+1} = R^{pq}$, $\bar{B}_L = [b_{ij}]$ is obtained by adding to an all-zero matrix R^{pq} to i,s and s,j positions where s is obtained from the corresponding B_L. Successive $\bar{B}_u$ are defined from $\bar{B}_{u+1}$ and B_u by adding $\bar{b}_{ij}^{u+1}$ to an all-zero matrix in the i,s, and s,j positions where the value of s is obtained from B_1. When $\bar{B}_1$ is obtained, $\bar{B}_1$ gives the y_{ij} which min (7) and that will satisfy the given constraints (4),(5), and (6). $\bar{B}_{L+1}$ is identical with the requirement matrix. This is certainly sufficient to satisfy all requirements; if we let $y_{ij} = \bar{b}_{ij}^{u+1}$ for all i and j.

Since we want to find the cheapest feasible network, we try to find a series of arcs which form a path from N_i to N_j and of minimum total cost. $\bar{B}_L$ is obtained by adding R^{pq} to a zero matrix to the two positions in $\bar{B}_L$ that in B_L indicate the intermediate node in which the minimum cost is achieved. As the minimum cost path for each R^{pq} consists of 2^L arcs or less, after tracing back from $\bar{B}_{L+1}$ L times to $\bar{B}_1$, $\bar{B}_1$ gives the y_{ij} which form the cheapest cost feasible network. The above algorithm gives the network which contains all the shortest paths, but does not give any information about whether one arc is used in a given shortest path (or requirement R^{pq}). This is due to y_{ij} being the sum of many requirements.

To convert the above algorithm into one that reveals the actual arcs used in different R^{pq}, we need only artificially give R^{pq} as follows. Let $R_1, R_1, \ldots, R_m$ be the given m requirements between m pairs of nodes. We shall let (8) $\sum_{i=1}^{k} R_i < R_{k+1}$ for $k = 1, \ldots, m-1$. For example,

$R_k = 2^{k-1}$ will satisfy (8).

When $\bar{B}_1 = y_{ij}$ is obtained, we can uniquely decompose $y_{ij} = \Sigma\delta_{fg}2^{k-1}$ where $\delta_{fg} = 1$ or zero. Suppose we want to find the shortest path from N_p to N_q ; first find $R^{pq} = R_k$, and then start from N_p . Following the arcs with 2^{k-1} in the decomposition of y_{ij} will lead N_p to N_q along the shortest path.

Take a five-node network as an example. The distances between nodes are shown in Table 1. (Note distances are not symmetric.) The D^2 together with B_1 matrix is shown in Table 2. The D_4 together with B_2 matrix is shown in Table 3.

	①	②	③	④	⑤
①	0	1	∞	6	∞
②	4	0	2	∞	∞
③	∞	∞	0	2	∞
④	∞	∞	∞	0	0
⑤	2	5	∞	∞	0

Table 1 Values of $[d_{ij}]$

	①	②	③	④	⑤
①	0	1	3	6	6
②	4	0	2	4	∞
③	∞	∞	0	2	2
④	2	5	∞	0	0
⑤	2	3	7	8	0

$[d_{ij}]^2 = D^2$

	①	②	③	④	⑤
①	X	2	2	4	4
②	1	X	3	3	5
③	1	2	X	4	4
④	5	5	3	X	5
⑤	1	1	2	1	X

B_1

Table 2

	①	②	③	④	⑤
①	0	1	3	5	5
②	4	0	2	4	4
③	4	5	0	2	2
④	2	3	5	0	0
⑤	2	3	5	7	0

$[d_{ij}]^4 = D^4$

	①	②	③	④	⑤
①	0	2	3	2	3
②	1	0	3	4	3
③	4	5	0	4	5
④	1	1	1	0	5
⑤	1	2	2	2	0

P^2

Table 3

Suppose that we want to find the shortest paths from N_1 to N_3, N_1 to N_4, N_2 to N_5, N_3 to N_5 and N_4 to N_1. Then we artificially set the corresponding flow requirements to be 1, 2, 4, 8, and 16. This is shown in Table 4.

	①	②	③	④	⑤
①	0	0	1	2	0
②	0	0	0	0	4
③	0	0	0	0	8
④	16	0	0	0	0
⑤	0	0	0	0	0

Table 4

From B_2 in Table 3 and $\bar{B}_3$ in Table 4, we get Table 5 (by adding R^{pq} to i, s and s, j).

	①	②	③	④	⑤
①	0	2	1	0	0
②	0	0	4	2	0
③	0	0	0	0	12
④	16	0	0	0	0
⑤	0	0	0	0	0

$\bar{B}_2$

Table 5

From B_1 in Table 2 and $\bar{B}_2$ in Table 5, we set $\bar{B}_1$ in Table 6.

	①	②	③	④	⑤
①	0	3	0	0	0
②	0	0	7	0	0
③	0	0	0	14	0
④	0	0	0	0	28
⑤	16	0	0	0	0

$\bar{B}_1$

Table 6

Note that every number in Table 6 has a unique decomposition as $\sum_{k=0}^{4} \delta_{fg} 2^k$. For example, if we want to find the shortest path from N_2 to N_5, then we look through $\bar{B}_1$ and find out what numbers contain 4 as their partial sum. In the present example, we have $7 = 4 + 2 + 1$, $14 = 8 + 4 + 2$ and $16 + 8 + 4$. Therefore, the shortest path is from N_2 to N_3, N_3 to N_4, and N_4 to N_5. If numbers are represented in the binary system, the process of checking whether a number contains 2^k is a very easy job.

We have given a method of calculating the shortest paths between all pairs of nodes in a network in which distances on arcs are given. Now we consider the problems of realization and synthesis. (The problems of realization and synthesis of undirected networks were studied by Hakimi and Yau.

Problem of realization: Given an nxn matrix $[\ell_{ij}]$. What are the conditions on $[\ell_{ij}]$ such that there exists an n-node network whose shortest distances are the given $[\ell_{ij}]$? It is easily seen that the necessary and sufficient conditions on ℓ_{ij} is that $\ell_{ik} \leq \ell_{ij} + \ell_{jk}$ for all i, j, k. The necessity is clear. If $\ell_{ik} > \ell_{ij} + \ell_{jk}$, then the shortest path from N_i to N_j of distance ℓ_{ij} plus the shortest path from N_j to N_k of distance ℓ_{jk} is a path from N_i to N_k and of less distance than ℓ_{ik}. This would contradict that ℓ_{ik} is the shortest distance from N_i to N_k.

The sufficient part is also clear. If $\ell_{ik} \leq \ell_{ij} + \ell_{jk}$ for all i, j, k, we can then construct $n(n-1)$ arcs between an n-node network with $d_{ij} = \ell_{ij}$ for all i, j. Then the shortest path from any node N_i to any other node N_j is just the single arc A_{ij} with distance d_{ij}.* Since there are many n-node networks that will have shortest distances equal to the prescribed $[\ell_{ij}]$, the following synthesis problem arises naturally. Given an nxn matrix $[\ell_{ij}]$, find a n-node network with given $[\ell_{ij}]$ as shortest distances, such that the total amount of d_{ij} associated with arcs of the network is minimum (i.e. min $\Sigma\, d_{ij}$). From now on, we assume $d_{ij} > 0$.

We shall use the words "optimum network" to mean the network with total d_{ij} minimum and satisfying the prescribed $[\ell_{ij}]$ as shortest distances. If an arc A_{ik} is in an optimum network, then the shortest path from N_i to N_k consists of the single arc A_{ik} with $d_{ik} = \ell_{ik}$. For, if ℓ_{ik} were equal to $d_{i1} + d_{12} + \ldots + d_{mk} < d_{ik}$, then any shortest path using A_{ik} can use the arcs $A_{j1}, A_{12}, \ldots, A_{mk}$

* This is the same condition given by Hakimi and Yau.

instead so that the arc A_{ik} can be replaced to reduce the total Σd_{ij} of the network. This would contradict the network is an optimum network.

Therefore, if an arc A_{ik} is in an optimum network, then $\ell_{ik} = d_{ik}$; if an arc A_{ik} is not in an optimum network, then

$$\ell_{ik} = \ell_{i1} + \ell_{12} + \ldots + \ell_{mk} \tag{9}$$

where $A_{i1}, A_{12}, \ldots, A_{mk}$ are arcs of the optimum network. It follows from (9) that

$$\ell_{ik} > \max(\ell_{i1}, \ell_{12}, \ldots, \ell_{mk}) \tag{10}$$

From (9), the arcs with d_{ij} equal to the smallest ℓ_{ij} must be in an optimum network.

If we know two arcs A_{ij}, A_{jk} are in any optimum network, then $\ell_{ik} \leq \ell_{ij} + \ell_{jk} = d_{ij} + d_{jk}$. If $\ell_{ik} = d_{ij} + d_{jk}$, then the arc A_{ik} will not be in any optimum network. If $\ell_{ik} < d_{ij} + d_{jk}$, then A_{ik} may or may not be in an optimum network. This depends on if there are other arcs such that $\ell_{ik} = d_{ip} + d_{pk}$. However, if $d_{ij} \leq d_{im}$ and $d_{jk} \leq d_{nk}$ for any nodes m, n, then $d_{ij} + d_{jk} \leq d_{im} + d_{mk}$ for any node m. Consequently, if $\ell_{ik} < d_{ij} + d_{jk} \leq d_{im} + d_{mk}$ then the arc A_{ik} is in an optimum network.

From the above considerations, we can develop the following algorithm for constructing an optimum network. Given the prescribed distance matrix $[\ell_{ij}]$, we shall circle some entries ℓ_{ij} and put brackets on other entries.

Step 1: Circle the smallest ℓ_{ij} which has not yet been circled, bracketed, or crossed out. (Intially, none of ℓ_{ij} has been circled, bracketed, or crossed out.) When an ℓ_{ij} has been circled, cross out the ith column and the jth row. Continue until every entry is circled or crossed out.

Step 2: Form the matrix D with d_{ij} equal to circled or bracketed ℓ_{ij}, $d_{ij} = \infty$ otherwise.

Step 3: Square D in the sense of (3). Let $D^2 = [d_{ij}'']$. Compare $[d_{ij}'']$ with $[\ell_{ij}]$. If ℓ_{ij} is not circled and equal to d_{ij}'', put a bracket on the ℓ_{ij}.

Step 4. If every ℓ_{ij} is either circled or bracketed, go to Step 5. Otherwise, erase all crossed out columns and rows and return to Step 1.

Step 5: The matrix with d_{ij} = circled ℓ_{ij} is the optimum network.

Note in Step 1. $[\ell_{ij}]$ is being circled with increasing magnitude. Because of the cross-out process, the $[\ell_{ij}]$ circled in the same step is not connected. From (9) and (10) they must be in the optimum network. In step 3, we check whether or not some of those circled arcs form a path and if they would imply the omission of some other arcs. Since we only square the distance matrix formed in Step 2, we only check paths consisting of two arcs. When an entry $[\ell_{ik}]$ is equal to $d_{ij} + d_{jk}$, this ℓ_{ik} is bracketed and will be used to form the matrix D in Step 2 of the next cycle. As at the end of Step 4 we erase all crossed out columns and rows, every entry ℓ_{ik} will sooner or later by circled or bracketed which

implies that arc is in or is not in the optimum network. The uniqueness of the optimum network follows from the observation that there is no alternate choice of circling an arc.

REFERENCES

1. R. Bellman, "On a Routing Problem", Quart. Appl. Math. Vol. 16, (1958), pp. 87-90.

2. C. Berge, "Theory of Graphs", (1958), Dunod, Paris.

3. G. B. Dantzig, "On the Shortest Route through a Network", Management Science, Vol. 6, No. 2, Jan. 1960.

4. L. R. Ford, Jr., "Network Theory", RAND Report, p. 923, July 1956.

5. R. E Gomory and T. C. Hu, "Multi-terminal Network Flows", J. of SIAM, Vol. 9, pp. 551-570, Dec. (1961).

6. R. E. Gomory and T. C Hu, "Synthesis of a Communication Network", J. of SIAM, Vol. 12, No. 2, June (1964), pp. 348-369.

7. S. L. Hakimi and S S. Yau, "Distance Matrix of a Graph and its Realizability." Quart of Appl Math., pp 305-318, Jan. 1965

8. E. F. Moore, "The Shortest Path through a Maze", proceedings of an International Symposium on the Theory of Switching, Part II, The Computation Laboratory of Harvard University Annal, Vol. 30, (1959), pp. 285-292.

9. M. Pollack and W. Wiebenson, "Solutions of the Shortest-route Problem - A Review", Operations Research 8, (1960), pp. 224-230.